Apellido y Nombre:	
Carrera:	DNI:
[Llenar con letra mayúscu	la de imprenta GRANDE]

Universidad Nacional del Litoral Facultad de Ingeniería y Ciencias Hídricas Departamento de Informática Algoritmos y Estructuras de Datos

Algoritmos y Estructuras de Datos. 1er Parcial. [2012-09-13]

ATENCIÓN: Para aprobar deben obtener un **puntaje mínimo** del 50 % en clases (Ej 1), 40 % en programación (Ej 2), y un 60 % sobre las preguntas de teoría (Ej 3).

[Ej. 1] [clases (30pt)]

- a) [lista (20pt)] Escribir la implementación en C++ del TAD lista (clase list) implementado por celdas (simple o doblemente) enlazadas por punteros ó cursores. (Indicar claramente qué implementación elige.) Los métodos a implementar son insert(p,x), erase(p), iterator::operator++(int) (postfijo), iterator::operator++() (prefijo). Atención: deben declarar los datos miembros de las clases a declarar o implementar.
- b) [pila-cola (10pt)] Escribir la implementación en C++ de los métodos push, pop, front, y top de los TAD pila y cola (clases stack y queue), según corresponda.
- [Ej. 2] [Programación (total = 50pt)] Recordar que en los ejercicios de programación deben usar la interfaz STL.
 - a) [is-sublist (20 puntos)]. Escribir una función bool is_sublist(list<int> &L1, list<int> &L2, listlistlist<int>::iterator> &iters) que dadas dos listas listas de enteros L1,L2 determina si la lista L2 es una sublista de L1, es decir que L2 se puede obtener a partir de L1 sólo eliminando algunos de sus elementos. Por ejemplo si L1=(3,2,6,5,4,1,8) y L2=(3,6,5,1), entonces is_sublist(L1,L2,iters) debe retornar true, mientras que si L2=(3,6,1,5) entonces debe retornar false ya que los elementos 1 y 5 están invertidos con respecto a su posición en L1. Adicionalmente, en el caso que SI sea sublista debe retornar por iters las posiciones (iteradores) que corresponden a los elementos de L1 que están en L2. En el ejemplo L1=(3,2,6,5,4,1,8) y L2=(3,6,5,1) debe retornar true y por iters 4 posiciones en la lista L1 correspondientes a los elementos 3,6,5,1.

RESTRICCIONES: el algoritmo debe ser O(n)

AYUDA: Para cada elemento de L1 verificar si se encuentra en L2 siempre partiendo desde la última posición en que se encontró el elemento anterior.

- b) [son-iguales (10 puntos)]. Escribir una función bool son_iguales(stack<int> &S, queue<int> &Q); que determina si el contenido de los contenedores S y Q son iguales (los elementos son idénticos). Después de terminar, los contenedores deben quedar en el mismo estado. Se pueden utilizar contenedores auxiliares.
- c) [submap (20 puntos)]. Dada una correspondencia map<string, string> M, y una clave k0, se desea extraer todos los pares de asignación correspondientes a claves conectadas con k0. Es decir, a partir de k0 se puede generar una secuencia L=(k0,k1,....) de la siguiente forma k1 = M(k0), k2=M(k1), hasta que o bien k_j no pertenece a las claves de M o bien k_j ya fue visitado. Se quiere extraer de la correspondencia aquellos pares de asignación cuyas claves están en la secuencia L. Por ejemplo si M1={(a,e),(c,a),(d,h),(e,g),(f,c),(g,d)} y k0=a entonces debe dejar M2={(a,e),(d,h),(e,g),(g,d)}

CONSIGNA: Escribir una funcion void submap(map<string, string> &M1, map<string> &M2, string k0); que deja en M2 los pares de asignación de aquellas claves que están en la secuencia que se genera por el algoritmo previo.

[Ej. 3] [Preguntas (total = 20pt, 4pt por pregunta)]

a) Ordenar las siguientes funciones por tiempo de ejecución. Además, para cada una de las funciones T_1, \ldots, T_5 determinar su velocidad de crecimiento (expresarlo con la notación $O(\cdot)$).

1er Parcial. [2012-09-13]

$$T_1 = 10n^3 + 2n! + 3\log_2 n + 4,$$

$$T_2 = 23! + 5n^4 + 2^n,$$

$$T_3 = n^3 + 3^n + 10,$$

$$T_4 = 2 + 5\log n,$$

$$T_5 = n + 2n^2 + 5n^3,$$

- b) Defina que entiende por un **algoritmo de búsqueda exhaustiva** y un **algoritmo ávido**. Explique como se aplicarían estas estrategias a un problema como el de coloración de grafos o el del agente viajero (TSP).
- c) ¿Como se define la **notación asintótica** T(n) = O(f(n))? ¿Porqué decimos que $(n+1)^2 = O(n^2)$ si siempre es $(n+1)^2 > n^2$?
- d) Si se implementa una **pila** en términos de **listas simplemente enlazadas**: ¿Dónde ubicaría el **tope** de la pila? ¿En el comienzo o en el fin de la lista? ¿Porqué?
- e) Discuta como se calcula el tiempo de ejecución de un bloque if en términos de los tiempos de ejecución del condicional $T_{\rm cond}$ y los bloques true y false: $T_{\rm true}$ y $T_{\rm false}$.

```
if (COND) {
   BODY_TRUE
} else {
   BODY_FALSE
}
```

1er Parcial. [2012-09-13]